ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа №3

Перегрузка операторов

Вариант №15

Выполнил студент группы РИС-23-3Б

Блинов А. Е.

Проверила доцент кафедры ИТАС

О. А. Полякова

2024 г.

**Постановка задачи**

1. Определить пользовательский класс.

2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

3. Определить в классе деструктор.

4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).

5. Перегрузить операцию присваивания.

6. Перегрузить операции ввода и вывода объектов с помощью потоков.

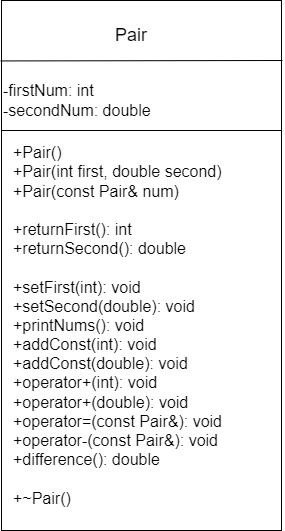
7. Перегрузить операции, указанные в варианте.

8. Написать программу, в которой продемонстрировать создание объектов и работу всех перегруженных операций.

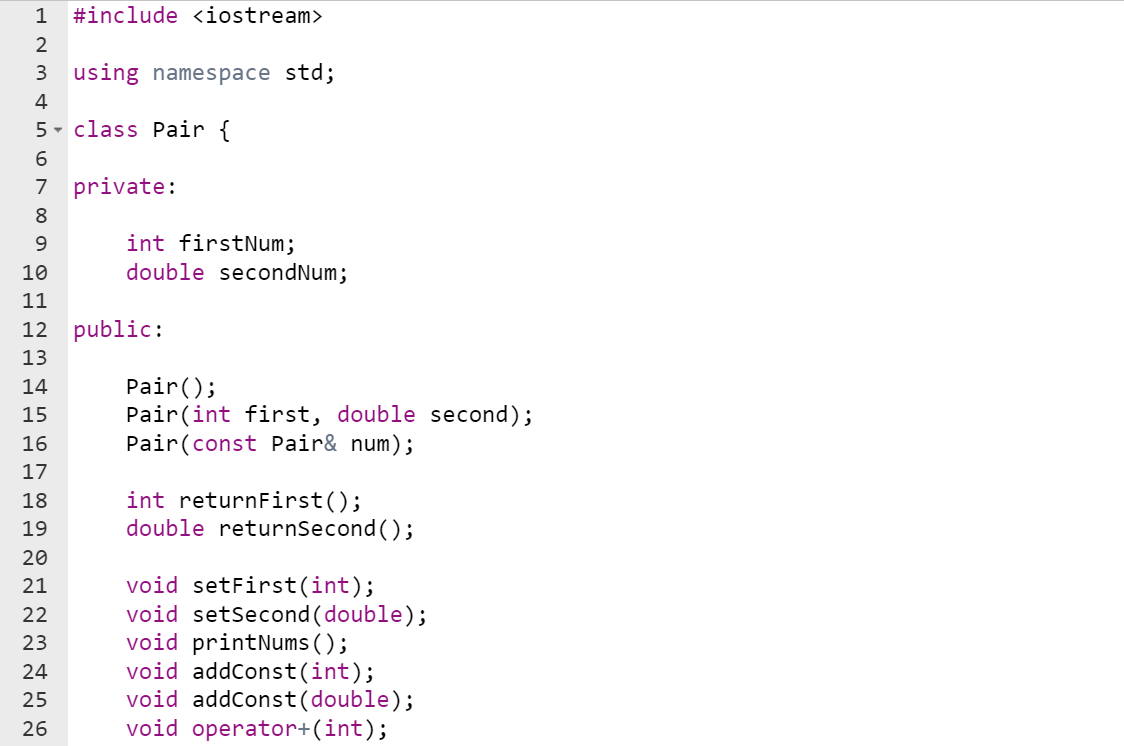
Создать класс Pair (пара чисел). Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

Реализовать: «-» - вычитание пар чисел; «+» - добавление константы к паре (увеличивается первое число, если константа целая, второе, если константа вещественная).

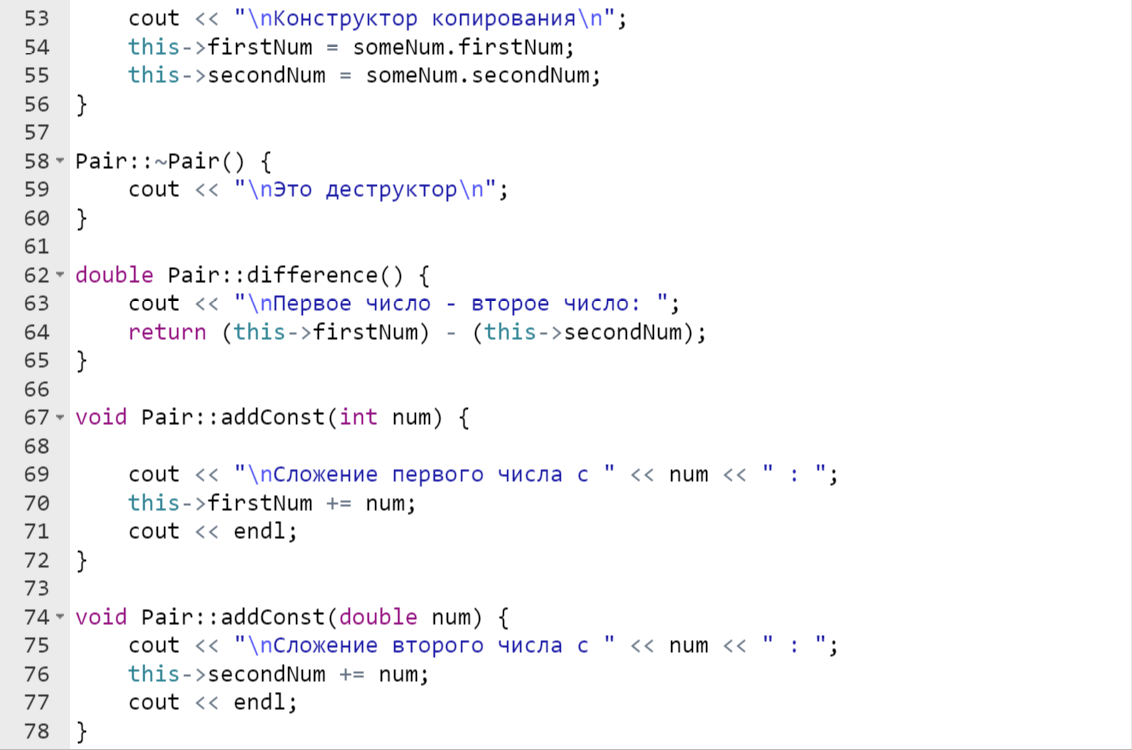
**UML диаграмма**

****

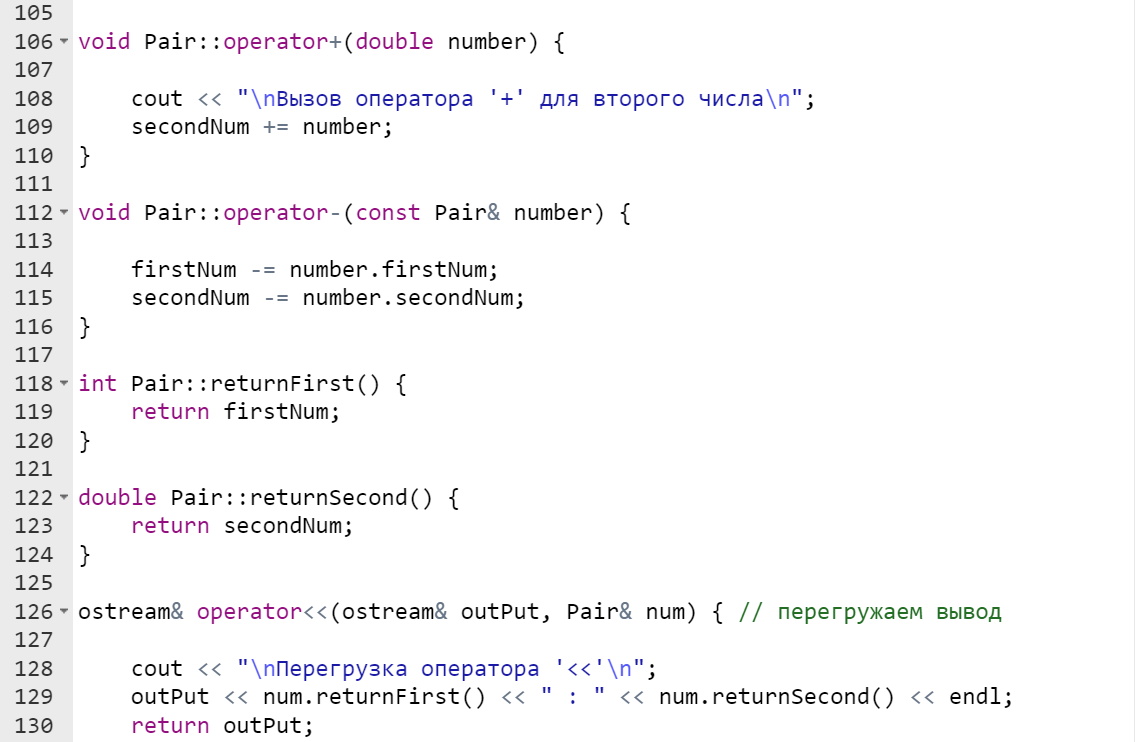
**Код на языке C++**

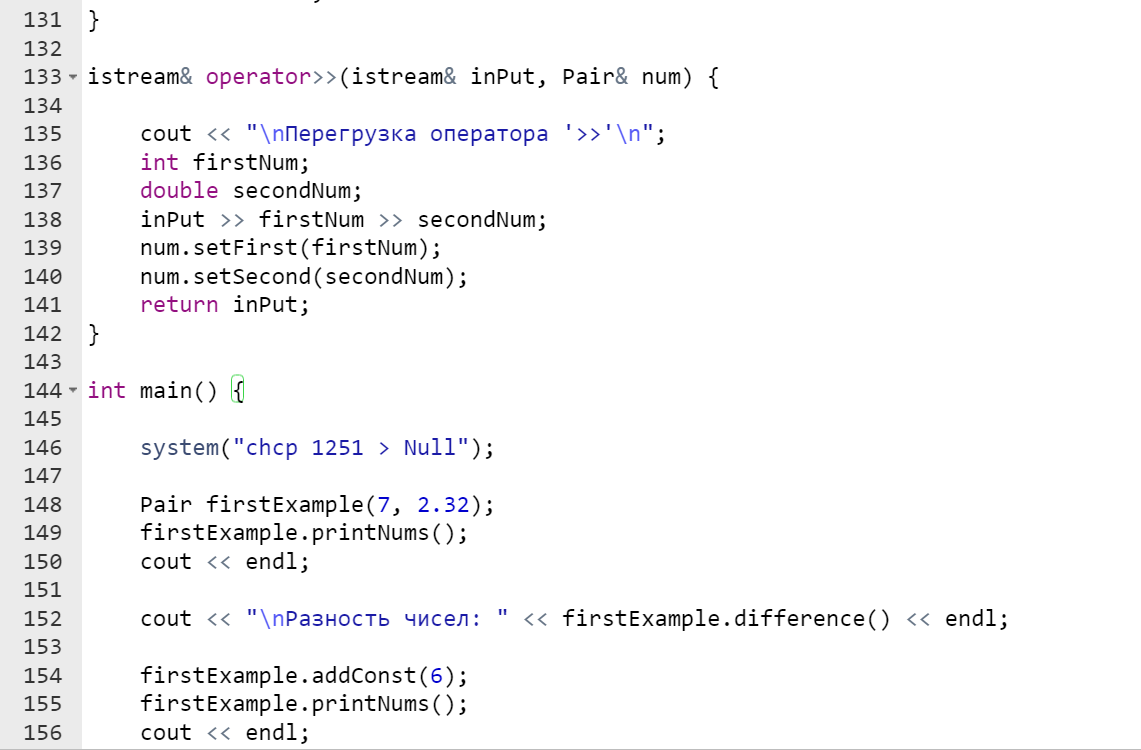


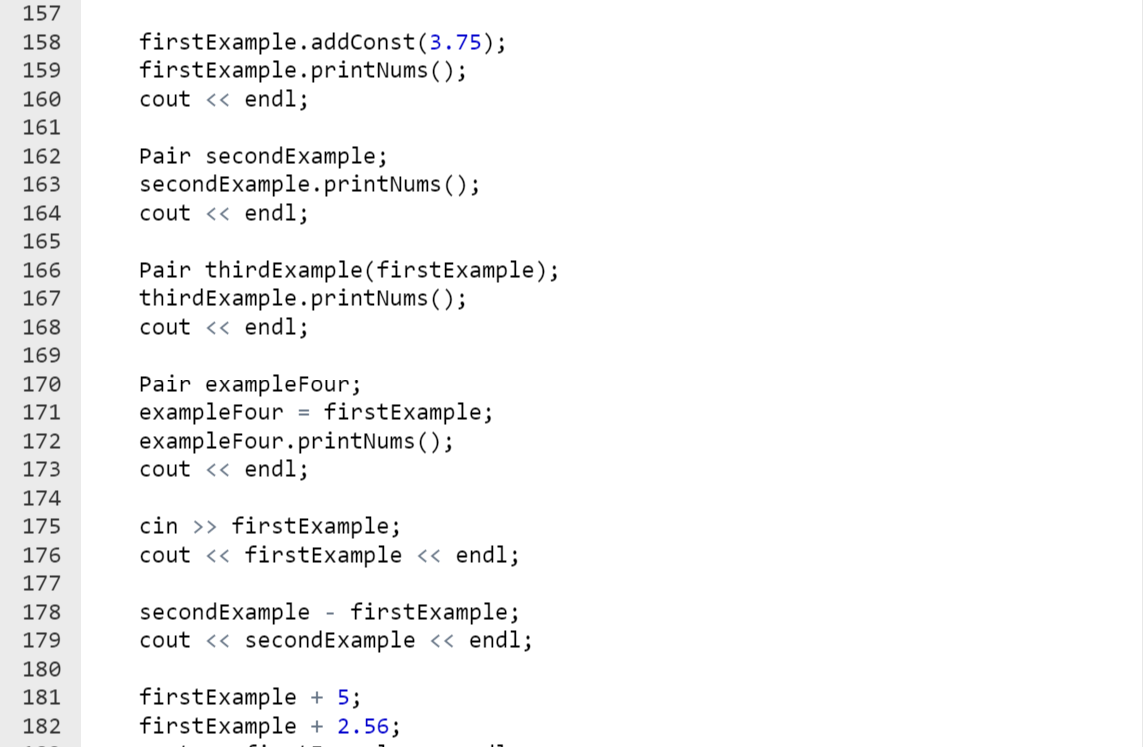


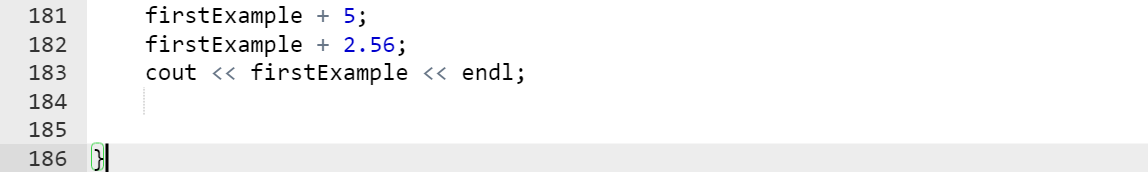




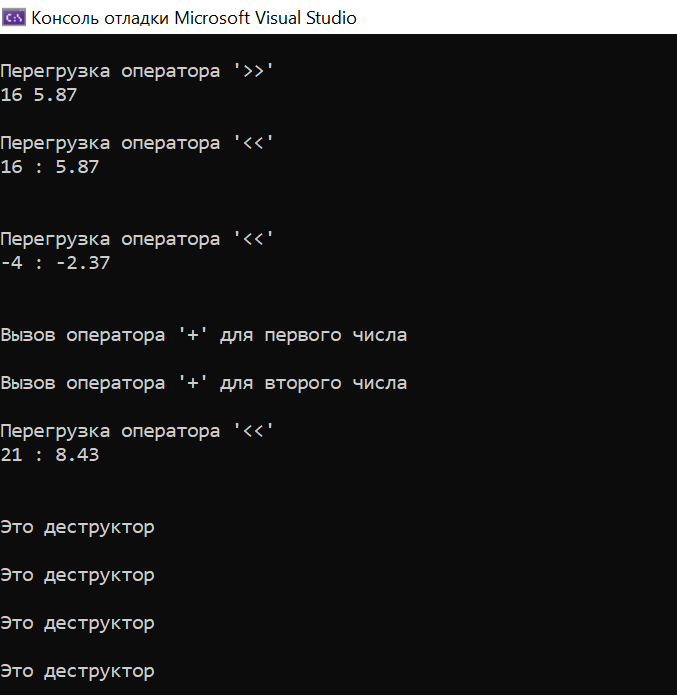








**Результаты работы программы**

****

**Выводы**

Программа успешно работает.

GitHub: <https://github.com/alexblin123/Laba-3-OOP.git>

**Контрольные вопросы**

1. **Для чего используются дружественные функции и классы?**

Дружественная функция – это функция, которая, не являясь компонентом класса, имеет доступ к его защищенным и собственным компонентам.

Класс может быть дружественным другому классу. Это означает, что все методы класса являются дружественными для другого класса. Дружественный класс должен быть определен вне тела класса, «предоставляющего дружбу».

1. **Сформулировать правила описания и особенности дружественных функций**

Такая функция должна быть описана в теле класса со спецификатором **friend.**

Не может быть компонентной функцией того класса, по отношению к которому определяется как дружественная; может быть глобальной функцией; может быть компонентной функцией другого ранее определенного класса.

1. **Каким образом можно перегрузить унарные операции?**

Два случая:

А) как компонентная функция

тип operator знак оператора( )

{

Действия…

return тип;

}

Б) как глобальная функция

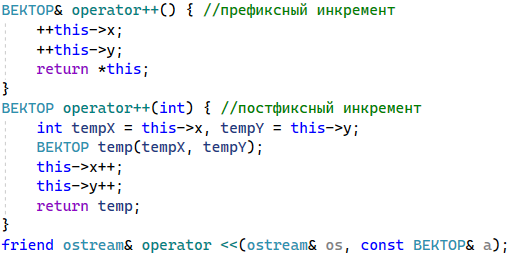
тип operator знак оператора(class&)

{

Действия…

return тип;

}



**4. Сколько операндов должна иметь унарная функция-операция, определяемая внутри класса?**

Унарные операции, перегружаемые в рамках определенного класса, могут перегружаться только через нестатическую компонентную функцию без параметров. Вызываемый объект класса автоматически воспринимается как операнд.

**5. Сколько операндов должна иметь унарная функция-операция, определяемая вне класса?**

Унарные операции, перегружаемые вне области класса (как глобальные функции), должны иметь один параметр типа класса. Передаваемый через этот параметр объект воспринимается как операнд.

**6. Сколько операндов должна иметь бинарная функция-операция, определяемая внутри класса?**

Операции, перегружаемые внутри класса, могут перегружаться только нестатическими компонентными функциями с параметрами. Вызываемый объект класса автоматически воспринимается в качестве первого операнда.

**7. Сколько операндов должна иметь бинарная функция-операция, определяемая вне класса?**

Операции, перегружаемые вне области класса, должны иметь два операнда, один из которых должен иметь тип класса.

**8. Чем отличается перегрузка префиксных и постфиксных унарных операций?**

Унарные операции инкремента ++ и декремента -- существуют в двух формах: префиксной и постфиксной. В современной спецификации С++ определен способ, по которому компилятор может различить эти две формы. В соответствии с этим способом задаются две версии функции operator ++() и operator --(). Они определены следующим образом:

Префиксная форма:

operator ++();

operator --();

Постфиксная форма:

operator ++(int);

operator --(int);

Указание параметра int для постфиксной формы не специфицирует второй операнд, а используется только для отличия от префиксной формы.

**9. Каким образом можно перегрузить операцию присваивания?**

Операция отличается тремя особенностями:

* операция не наследуется;
* операция определена по умолчанию для каждого класса в качестве операции поразрядного копирования объекта, стоящего справа от знака операции, в объект, стоящий слева.
* операция может перегружаться только в области определения класса.

Пример: void Pair::operator=(const Pair& number) {

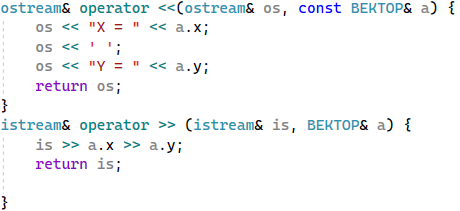
This->объект = объект

… }

**10. Что должна возвращать операция присваивания?**

Функция operator= возвращает не объект, а ссылку на него. (\*this)

**11. Каким образом можно перегрузить операции ввода-вывода?**

****